REMARKS

Claims 1-21 are now pending in the application. The Examiner is respectfully requested to reconsider and withdraw the rejection(s) in view of the remarks contained herein.

REJECTION UNDER 35 U.S.C. § 102

Applicant respectfully traverses the rejection of Claims 1-6, 8-9, 11-13, and 15-20 under 35 U.S.C. § 102(e) as being anticipated by Logan, III et al. (U.S. Pat. No. 6,243,857).

Referring to Claim 1, Logan, III et al. do not show, teach, or suggest a reason code module associated with a flowcharting module that assigns first and second reason codes to first and second flowchart blocks to capture operational modes of a process.

Logan, III et al. teach a machine control system that includes a multi-block flowchart. The multi-block flowchart represents a program, and a computer compiles the program from the flowchart to control operations of a machine. (Abstract). The system also includes a debugger for displaying the flowchart in a debugger window for runtime execution control of the program. During a debugging phase, the debugger takes control over the machine control system upon an interrupt of the execution of the flowchart. (col. 4, line 1). Flowchart blocks being executed at the time of the interrupt are highlighted and/or listed to allow a user to change values of the flowchart blocks.

The debugger does not assign reason codes to flowchart blocks, as required by the claims. As with conventional debugging programs in computer programming, the debugger identifies errors in the flowchart program during the debugging phase. This gives a programmer an opportunity to correct problems in the initial program. (col. 4, line 23). Furthermore, operational modes of the process are not captured, as required by the claims. For example, the machine control system taught by Logan, III et al. does not monitor a relative frequency that a particular flowchart block is executed.

In the Response to Arguments, the Examiner contends that on page 7, lines 7-13, Applicants teach a reason code as a status of a process in a flowchart that is described by action or decision blocks. **Final Office Action, p. 2 (October 10, 2003).** Applicants disagree. On page 7, line 7, Applicants teach that a reason code module allows a user to *assign reason code to the status of* a process described by action or decision blocks in a flowchart. (Emphasis Added). On page 2, line 20, Applicants teach that reason codes that are assigned to flowchart blocks provide information that identifies the reasons for process downtime.

Errors that are identified in the machine control system that is taught by Logan, III et al. are errors that the debugger detects in the flowchart code. (col. 3, line 65). The errors are not occurrences or reasons for downtime in the actual process that is being furthered by the flowchart. As long as errors are identified by the debugger in Logan, III et al., the process that is furthered by the flowchart is idled or delayed.

Applicants utilize reason codes that are assigned to flowchart blocks to capture operational modes of a process. After a reason code is assigned to a flowchart block, the reason code is generated during execution of the flowchart. The reason code is generated when the block that it is assigned to is executed. When a performance analysis module records the generation of the reason codes, the reasons for process downtime may be determined based on the relative occurrence of particular reason codes. (page 7, line 12).

For example, the generation of reason codes that are assigned to blocks 306 and 318 in Figure 5 do not indicate an error in the flowchart program. They indicate the execution of the block that they are assigned to. Generation of a reason code that is assigned to block 306 in Figure 5 indicates up time of the process. (page 12, line 9). Generation of a reason code that is assigned to block 318 of Figure 5 indicates a shutdown of the process due to a malfunctioning furnace. (page 12, line 11). If a performance analysis module detects that the reason code assigned to block 318 is generated far more often than the reason code assigned to block 306, the reason for process downtime may be linked to a malfunctioning furnace. However, generation of the reason code assigned to block 318 does not indicate an error in the flowchart code. It simply indicates when block 318 is executed by the flowchart.

The Examiner contends that the code "air pressure is too low, then turn on" is interpreted as reason code according to the specification and claim language. **Final Office Action at 3.** Applicant disagrees. The fact that Logan, III et al. illustrates a scenario where air pressure is too low and something is turned on in response to this does not indicate that the process experiences downtime. For example, it may be desirable or an expected occurrence for the pressure to drop at a certain point in the process illustrated by Logan, III et al.

This illustrates the advantage of the reason code module as claimed by Applicants. The reason code module assigns reason codes to specific blocks in a flowchart that are capable of indicating process downtime. (page 6, line 20). Furthermore, Logan, III et al. do not suggest monitoring the relative frequency of execution of a particular block to indicate process downtime. For example, even if the occurrence of the code "air pressure is too low, then turn on" is undesirable, the code may only be generated once in a very long period of time and may not indicate significant process downtime.

Claims 2-10 depend directly or indirectly from Claim 1 and are allowable over Logan, III et al. for the same reasons.

Referring now to Claim 11, Logan, III et al. do not show, teach, or suggest a reason code module associated with a flowcharting module that assigns first and second reason codes to first and second flowcharting blocks to capture operational modes of a process. Logan, III et al. also does not show, teach, or suggest flowchart object code that is executed by a flowchart run time engine and that generates the first reason code during execution of the first flowchart block in the flowchart object code.

As discussed above, Logan, III et al. teach a debugger in a flowchart program that is used for displaying the flowchart in a debugger window for runtime execution control of the program. During a debugging phase, the debugger takes control over the machine control system upon an interrupt of the execution of the flowchart. (col. 4, line 1). Flowchart blocks being executed at the time of the interrupt are highlighted and/or listed to allow a user to change values of the flowchart blocks.

The debugger does not assign reason codes to flowchart blocks, as required by the claims. The debugger identifies errors in the flowchart program during the debugging phase. Furthermore, operational modes of the process are not captured, as required by the claims. For example, the machine control system taught by Logan, III et al. does not monitor a relative frequency that a particular flowchart block is executed. Errors that are identified in the machine control system that is taught by Logan, III et al. are errors that the debugger detects in the flowchart code. (col. 3, line 65).

As taught by Applicants, the reason code module assigns reason codes to specific blocks in a flowchart that are capable of indicating process downtime. (page 6, line 20). Furthermore, Logan, III et al. do not suggest monitoring the relative frequency of execution of a particular block to indicate process downtime.

Claims 12-14 depend directly or indirectly from Claim 11 and are allowable over Logan, III et al. for the same reasons.

Referring to Claim 15, Logan, III et al. does not show, teach, or suggest assigning first and second reason codes to a process by first and second flowchart blocks in a flowchart source code to capture operational modes of the process.

As discussed above, Logan, III et al. teach a debugger in a flowchart program that is used for displaying the flowchart in a debugger window for runtime execution control of the program. Flowchart blocks being executed at the time of an interrupt are highlighted and/or listed to allow a user to change values of the flowchart blocks. The debugger does not assign reason codes to flowchart blocks, and operational modes of the process are not captured, as required by the claims. Errors that are identified in the machine control system that is taught by Logan, III et al. are errors that the debugger detects in the flowchart code. (col. 3, line 65).

Claims 16-21 depend directly or indirectly from Claim 15 and are allowable over Logan, III et al. for the same reasons.

CONCLUSION

It is believed that all of the stated grounds of rejection have been properly traversed, accommodated, or rendered moot. Applicant therefore respectfully requests that the Examiner reconsider and withdraw all presently outstanding rejections. It is believed that a full and complete response has been made to the outstanding Office Action, and as such, the present application is in condition for allowance. Thus, prompt and favorable consideration of this amendment is respectfully requested. If the Examiner believes that personal communication will expedite prosecution of this application, the Examiner is invited to telephone the undersigned at (248) 641-1600.

Respectfully submitted,

Dated: 11/25/03

By: [illegible]
Michael D. Wiggins
Reg. No. 34,754

HARNESS, DICKEY & PIERCE, P.L.C.
P.O. Box 828
Bloomfield Hills, Michigan 48303
(248) 641-1600

MDW/SLS